Appendix

Below are the appendices of the paper "Modeling and Verifying WNT Signaling Pathway".

A The picture of WNT signaling pathway

WNT signaling plays a significant role in cell development and oncogenesis [7][8][9][24]. A variety of extracellular, cytoplasmic, and nuclear components modulate WNT signaling procedure. The pathway is a conserved one against perturbations in various environment. It is required for adult tissue maintenance and other important functions of development. The pathway controls the concentration of β-catenin with the cooperation of other proteins and enzymes, to regulate the gene expression in the nucleus(a mathematical model is given in [9]). The expression then contributes to the cell's behavior, including multiple feedback loops to the steps of the signaling pathway. WNT malfunction is implicated in such forms as cancer(such as human cancer) and degenerative diseases.

The whole WNT signaling pathway is a rather complex and intricate picture. Below we present the canonical WNT signaling pathway graphically (Figure 1)(the source of the diagram is shown at the bottom of it), and explain it in more detail. The main line of the WNT signaling pathway can be described as:

1. WNT binds to receptor Frz
2. Frz reacts with Dsh
3. Dsh causes β-catenin to escape from being degraded
4. β-catenin accumulates and enters the nucleus
5. β-catenin regulates TCF/LEF and related tagert genes' expression

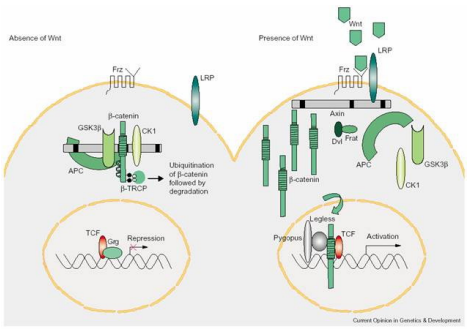

Figure 1: WNT signaling pathway

B A brief introduction to Maude

Maude is a mighty software for implementing various rewriting systems and other related systems, such as various internet computing, mobile computing, bio-informatics. We use Maude to implement the model of WNT signaling pathway. Here we give a brief introduction to Maude. More can be found on the web-site of Maude [11].

Maude implements two logics, membership equational logic and rewriting logic. A membership equational theory is a pair $(\Sigma, E \cup A)$. Σ is the signature, which specifies the type structure of *sorts, subsorts, kinds* and *overloaded operators*. E is the set of equations (possibly conditional) and membership declarations. A is the set of equational attributes (such as $assoc$(associative), $comm$(commutative)) for distinct operators. The deduction of equations comstitutes the computation of the theory. A rewriting theory is a 4-tuple $\mathcal{R} = (\Sigma, E \cup A, \phi, R)$. $(\Sigma, E \cup A)$ is the membership equation theory in the rewriting theory. ϕ is the function that specifies the frozen arguments of each operators in Σ. And R is the set of rewriting rules (possibly conditional). Rewriting rules deduction, mixed by equational deductions, forms the computation of the rewriting theory.

Maude is composed of modules, mainly two kinds, the functional module consisting of *equations*, and the system module consisting of *rules*. And the two kinds of modules correspond to membership

equational logic and rewriting logic, respectively. Below we describe the basic structure of a Maude program.

A typical Maude program comprises several parts, typically the following (we use examples to illustrate them):

- Sorts. The following statements define an array (of integers) in most programming languages.

$$\begin{array}{ll} op & nil : -> Array. \\ op & \_,\_ : Int\ Array -> Array. \end{array}$$

- Functions. The following statements declare two functions on an array.

$$\begin{array}{lll} op & size\_ : Array & -> Nat. \\ op & \_in\_ : IntArray & -> Bool. \end{array}$$

- Variables. The following statements declare three variables.

$$\begin{array}{ll} var & l : Array. \\ vars & i\ j : Int. \end{array}$$

- Equations. The following statements define the *size* function declared above.

$$\begin{array}{lll} eq & size(nil) = & 0. \\ eq & size(i, l) = & s\ size(l). \end{array}$$

- Objects and messages. In Maude, one can define objects and messages to them. To be concise, we work on instances. For example, the following is two objects consisting of the information of the salary of an employee in a certain company (you can view them as two records in the company's database), and two messages to modify them.

$$\begin{array}{c} \langle\ 'John : Employee \mid salary : 3000\ \rangle \\ \langle\ 'Mary : Employee \mid salary : 3500\ \rangle \end{array}$$

$$\begin{array}{c} promote('John, 500) \\ demote('Mary, 500) \end{array}$$

- Rules. Then the following two rules define the messaging on objects (Qid is the sort of quoted identifiers). One message is to increase salary and the other is to decrease salary.

$$\begin{array}{ll} rl & \langle\ i : Employee \mid salary : m\ \rangle\ promote(i, n) => . \\ & \quad \langle\ i : Employee \mid salary : (m + n)\ \rangle. \end{array}$$

$$\begin{array}{ll} crl & \langle\ i : Employee \mid salary : m\ \rangle\ demote(i, n) => . \\ & \quad \langle\ i : Employee \mid salary : (m - n)\ \rangle\ \text{if}\ m >= n. \end{array}$$

Two instances of possible computation of the partial program above are as follows.

$$\begin{array}{ll} & size(3, (8, (13, nil))) \\ = & s\ size((8, (13, nil))) \\ = & s\ s\ size(13, nil) \\ = & s\ s\ s\ size(nil) \\ = & s\ s\ s\ 0, \end{array}$$

where ($s\ s\ s\ 0$) indicates 3 in natural numbers (s can be seen as the successor function).

And the two objects evolve to

$$\begin{array}{c} \langle\ 'John : Employee \mid salary : 3500\ \rangle \\ \langle\ 'Mary : Employee \mid salary : 3000\ \rangle \end{array}$$

In our implementation of the WNT signaling pathway model, we use some of the mechanism of Maude to realize the 'rules' in the P system. And we may use some mechanism of Maude that is not mentioned above, in that case, we will make explicit explanation.

C LTL

Here we give a brief introduction to the logic used in characterizing the properties, that is, the LTL, short for linear temporal logic, which can be used to describe the reachability from an initial state of the system under inspection. It is a relatively simple logic that is widely used for its intuitive appeal and well-developed proof and decision methods. Given AP as the set of atomic propositions, propositional LTL formulae (LTL_{AP}) are defined as follows (suppose $\varphi, \psi \in LTL_{AP}$):

- Basic operators:
  1. True: $\top \in LTL_{AP}$
  2. Atomic propositions: $AP \subseteq LTL_{AP}$
  3. Boolean operators $\neg, \vee$: $\neg\varphi, \varphi \vee \psi \in LTL_{AP}$
  4. Next operator $\bigcirc$: $\bigcirc\varphi \in LTL_{AP}$
  5. Until operator $\mathcal{U}$: $\varphi\mathcal{U}\psi \in LTL_{AP}$
- Derived operators:
  1. Boolean operators:
     - False: $\bot \triangleq \neg\top$
     - Conjunction operator: $\varphi \wedge \psi \triangleq \neg((\neg\varphi) \vee (\neg\psi))$
     - Implication operator: $\varphi \rightarrow \psi \triangleq ((\neg\varphi) \vee \psi)$
  2. Temporal operators:
     - Eventual operator: $\diamondsuit \triangleq \top\mathcal{U}\varphi$
     - Henceforth operator: $\square\varphi \triangleq \neg\diamondsuit\neg\varphi$
     - Release operator: $\varphi\mathcal{R}\psi \triangleq \neg((\neg\varphi)\mathcal{U}(\neg\psi))$
     - Unless operator: $\varphi\mathcal{W}\psi \triangleq (\varphi\mathcal{U}\psi) \vee (\square\varphi)$
     - Leads-to operator: $\varphi \rightsquigarrow \psi \triangleq \square(\varphi \rightarrow (\diamondsuit\psi))$
     - Strong implication operator: $\varphi \Rightarrow \psi \triangleq \square(\varphi \rightarrow \psi)$
     - Strong equivalence operator: $\varphi \Leftrightarrow \psi \triangleq \square(\varphi \leftrightarrow \psi)$

The models of temporal logic are Kripke structures. A Kripke structure is a triple $\mathcal{A} = (A, \rightarrow_{\mathcal{A}}, L)$. A is the states set. $\rightarrow_A \subseteq A \times A$ is the transition relation. $L : A \longrightarrow \mathcal{P}(AP)$ is the labeling function, which associates a set of atomic propositions that hold in a state to that state. And we define $Path(\mathcal{A}_a)$ as the set of computation pathes starting from the state $a \in A$, where $Path(\mathcal{A})_a$ contains functions $\pi : \mathbb{N} \longrightarrow A$ of the form

$$\pi(0) = a \text{ and } \pi(n) \rightarrow_{\mathcal{A}} \pi(n+1)$$

for each $n \in \mathbb{N}$ ($\mathbb{N}$ is the set of natural numbers).

The elements of the satisfaction relation that defines the semantics of LTL is of the form

$$\mathcal{A}, a \models \varphi,$$

where $\mathcal{A}$ is a Kripke structure, a is a state in $\mathcal{A}$, and $\varphi \in LTL_{AP}$ is an LTL formula. The relation states that $\mathcal{A}, a \models \varphi$ holds if and only if the path satisfaction function

$$\mathcal{A}, \pi \models \varphi$$

holds for each path $\pi \in Path(\mathcal{A})_a$.

The path satisfaction relation can be defined as follows (we assume $\mathcal{A}$ is the Kripke structure, π is the path function, $p \in AP$, and φ, ψ are LTL_{AP} formulae):

- $\mathcal{A}, \pi \models \top$ holds always
- $\mathcal{A}, \pi \models p$ iff $p \in L(\pi(0))$
- $\mathcal{A}, \pi \models \neg\varphi$ iff $\mathcal{A}, \pi \not\models \varphi$
- $\mathcal{A}, \pi \models \varphi \vee \psi$ iff $\mathcal{A}, \pi \models \varphi$ or $\mathcal{A}, \pi \models \psi$
- $\mathcal{A}, \pi \models \bigcirc\varphi$ iff $\mathcal{A}, \pi \circ s \models \varphi$, where $s : \mathbb{N} \rightarrow \mathbb{N}$ is the successor function and $\circ$ is the function composition operation.
- $\mathcal{A}, \pi \models \varphi\mathcal{U}\psi$ iff
  There exists $n \in \mathbb{N}$ s.t. $(\mathcal{A}, \pi \circ s^n \models \psi)$ and for all $m \in \mathbb{N}$, $m < n$ implies that $\mathcal{A}, \pi \circ s^m \models \varphi$

More on LTL and its model checking method can be found in [3]. And more on Maude's module MODEL-CHECKER can be found in chapter 9 of [12].

D Implementation code of the WNT signaling pathway

Here we give the complete implementation code of the WNT signaling pathway.

WNT signaling pathway model implementation

```
mod WNT is
  sorts Obj Rule Mem Name ChannelState Channel .
  op n0 : -> Name .
  op n1 : -> Name .
  op n2 : -> Name .
  op c1 : -> Name .
  op c2 : -> Name .
  op s0 : -> ChannelState .
  op s1 : -> ChannelState .
  op t0 : -> ChannelState .
  op t1 : -> ChannelState .
  op wnt : -> Obj .
  op frz-f : -> Obj .
  op wnt~frz-f : -> Obj .
  op frz-tgr : -> Obj .
  op LRP-f : -> Obj .
  op wnt~wnt~LRP-f : -> Obj .
  op LRP-tgr : -> Obj .
  op frz-sf : -> Obj .
  op LRP-sf : -> Obj .
  op dsh : -> Obj .
  op dsh-tgr : -> Obj .
  op PP2A : -> Obj .
  op axin~gsk3B~apc~ck1 : -> Obj .
  op axin : -> Obj .
  op gsk3B : -> Obj .
  op apc : -> Obj .
  op ck1 : -> Obj .
  op B-catenin-tgr : -> Obj .
  op B-catenin : -> Obj .
  op TCF-LEF-tgr : -> Obj .
  op TCF-LEF~Gro : -> Obj .
  op TCF-LEF : -> Obj .
  op Gro : -> Obj .
  op B-catenin~TCF-LEF : -> Obj .
  op Legless : -> Obj .
  op Pygopus : -> Obj .
  op B-catenin~TCF-LEF~Legless~Pygopus : -> Obj .
  op TargetGene : -> Obj .
  op TargetGene-trans-activator : -> Obj .
  op TargetGene-expressed-protein : -> Obj .
  op config0 : -> Obj .
  op config01 : -> Obj .
  op config1 : -> Obj .
  op config2 : -> Obj .
  op ruleset0 : -> Rule .
  op ruleset1 : -> Rule .
  op ruleset2 : -> Rule .
  op channel01 : -> Channel .
  op channel12 : -> Channel .
  op [_:_,_,_] : Name ChannelState Name Name -> Channel .
  op empty : -> Channel .
```

```
  op __ : Channel Channel -> Channel [ assoc comm id: empty ] .
  op nul : -> Obj .
  op  none : -> Rule .
  op O : -> Mem .
  op init : -> Mem .
  op init1 : -> Mem .
  op __ : Obj Obj -> Obj [ assoc comm id: nul] .
  op <_,_> : Obj Obj -> Rule .
  op <_,Out,_> : Obj Obj -> Rule .
  op <_,_,In,_> : Obj Obj Name -> Rule .
  op <_:_,_/_,_> : Name ChannelState Obj Obj ChannelState -> Rule .
  op _|_ : Rule Rule -> Rule [ assoc comm id: none] .
  op [_:_,_,_,_] : Name Obj Rule Channel Mem -> Mem .
  op __ : Mem Mem -> Mem [ assoc comm id: O] .
  vars O1 O2 O3 O4 O5 : Obj .
  vars N1 N2 N3 : Name .
  vars M1 M2 M0 M3 : Mem .
  vars R1 R2 R3 : Rule .
  vars C1 C2 C3 : Channel .
  vars C1n C2n : Name .
  vars S1 S2 : ChannelState .

  rl [ rule1 ] :
     [N1:O1,<C1n:S1,O2/O3,S2>|R1,[C1n:S1,N2,N3] C1,[N2:O2 O4,R2,C2,
        M2] [N3:O5 O3,R3,C3,M3] M1]
     =>
     [N1:O1,<C1n:S1,O2/O3,S2>|R1,[C1n:S2,N2,N3] C1,[N2:O4 O3,R2,C2,
        M2] [N3:O5 O2,R3,C3,M3] M1].
  rl [ rule2 ] :
     [N1:O1 O2,<C1n:S1,O2/O3,S2>|R1,[C1n:S1,N1,N2] C1,[N2:O3 O4,R2,
        C2,M2] M1]
     =>
     [N1:O1 O3,<C1n:S1,O2/O3,S2>|R1,[C1n:S2,N1,N2] C1,[N2:O2 O4,R2,
        C2,M2] M1].
  rl [ rule3 ] :
     [N1:O1 O2,<O2,O3>|R1,C1,M1]
     =>[N1:O1 O3,<O2,O3>|R1,C1,M1].
  rl [ rule4 ] :
     [N1:O1 O2,<O2,O3,In,N2>|R1,C1,[N2:O4,R2,C2,M2] M1]
     =>
     [N1:O1,<O2,O3,In,N2>|R1,C1,[N2:O3 O4,R2,C2,M2] M1].
  rl [ rule5 ] :
     [N1:O1,R1,C1,[N2:O3 O4,<O3,Out,O2>|R2,C2,M2] M1]
     =>
     [N1:O1 O2,R1,C1,[N2:O4,<O3,Out,O2>|R2,C2,M2] M1].

eq ruleset0 = < c1 : s0 , frz-tgr / nul , s1 > | < c1 : s1 , LRP-tgr
/ nul , s0 > | < wnt frz-f , wnt~frz-f > | < wnt~frz-f , frz-tgr > |
< wnt LRP-f , wnt~wnt~LRP-f > | < wnt~wnt~LRP-f , LRP-tgr > . eq
ruleset1 = < c2 : t0 , B-catenin-tgr / nul , t1 > | < c2 : t1 ,
B-catenin / nul , t0 > | < frz-tgr , frz-sf > | < LRP-tgr , LRP-sf >
| < frz-sf LRP-sf dsh , dsh-tgr > | < dsh-tgr PP2A
axin~gsk3B~apc~ck1  , axin gsk3B apc ck1 B-catenin-tgr B-catenin > .
eq ruleset2 = < B-catenin-tgr , TCF-LEF-tgr > | < TCF-LEF-tgr
TCF-LEF~Gro , TCF-LEF Gro > | < B-catenin TCF-LEF ,
B-catenin~TCF-LEF > | < B-catenin~TCF-LEF Legless Pygopus ,
B-catenin~TCF-LEF~Legless~Pygopus > | <
```

```
B-catenin~TCF-LEF~Legless~Pygopus TargetGene ,
TargetGene-trans-activator > | < TargetGene-trans-activator ,
TargetGene-expressed-protein > .

endm

mod WNT-PREDS is
  protecting WNT .
  including SATISFACTION .
  subsort Mem < State .
  op HasObj : Obj -> Prop .
  op HasObj2 : Obj -> Prop .
  op Has : Obj -> Prop .
  vars O1 O2 O3 OT : Obj .
  vars N1 N2 N3 : Name .
  vars M1 M2 M0 M3 : Mem .
  vars R1 R2 R3 : Rule .
  vars C1 C2 C3 : Channel .
  eq [N1:O1 OT,R1,C1,M1] |= Has(OT) = true.
  eq [N1:O1,R1,C1,[N2:O2,R2,C2,[N3:O3 OT,R3,C3,M3] M2] M1]
     |= HasObj(OT) = true .
  eq [N1:O1,R1,C1,[N2:O2 OT,R2,C2,M2] M1] |= HasObj2(OT)=true .
endm

mod WNT-CHECK is
  including WNT-PREDS .
  including MODEL-CHECKER .
  including LTL-SIMPLIFIER .
eq config0 = wnt wnt frz-f LRP-f . eq config01 = wnt frz-f LRP-f .
eq config1 = dsh PP2A axin~gsk3B~apc~ck1 . eq config2 = TCF-LEF~Gro
Legless Pygopus TargetGene . eq init = [ n0 : config0 , ruleset0 , [
c1 : s0 , n0 , n1 ] , [ n1 : config1 , ruleset1 , [ c2 : t0 , n1 ,
n2 ] , [ n2 : config2 , ruleset2 , empty , 0 ] ] ] . eq init1 = [ n0
: config01 , ruleset0 , [ c1 : s0 , n0 , n1 ] , [ n1 : config1 ,
ruleset1 , [ c2 : t0 , n1 , n2 ] , [ n2 : config2 , ruleset2 , empty
, 0 ] ] ] . endm
```

E Future work

We think the work here can be furthered in at least two directions.

The first one is that the model can be extended. Since the WNT signaling pathway is a rather complicated process forming and residing in a larger network of signals and the model we construct and analyze above is a typical yet basic branch of the whole picture, we can add more related branches to the model, such as another sub-pathway of WNT signaling, for instance through the Ca^{+}, which may intersect with the canonical pathway to constitute a far more intricate network worth modeling and analyzing. Moreover, There are feedback loops in the WNT signaling pathway, that is, the expression of target genes will possibly influence other genes or proteins in the nucleus or the cytoplasm, and successively regulate the proteins relative to WNT signaling on the transmitting way, thus feedback modulating the whole procedure. The feedback loops are indispensable in living cells and taking them into our model in the future is a natural idea. We think there are at least two approaches to include the feedback mechanism in our model.

1. After the target genes are expressed, as described partially above in previous sections, their generated proteins interact with certain components on the WNT pathway to down-regulate/up-regulate them. And the regulating rule is not hard to design;

2. The feedback regulation is realized by controlling the secretion of the components on the WNT pathway in terms of the effect of WNT signalling on gene regulation.

The second approach need expanding our model since the secretion of each components on the WNT pathway is not included in the model. In contrast, the first approach is more direct. But excluding the secretion part from the model renders it inconsistent with the actual process, for example the cycle is not continuous. Hence our model is a simplified version, where the total amount of some proteins(such as WNT) will not exceed some constant value. But we think this should be a good and rigorous first step toward a more complete model. To some extent, the model in this paper can be thought of as a core of the whole WNT signalling pathway.

Another one is to apply parallelism to rules executing on channels[25]. This thought rises from the fact that the triggering of the WNT signaling, that is, the binding of the WNT protein to the receptors on the cell surface, is not in one single way, and can be fired in several locations of the cell surface and through different ways (i.e. by different cross-membrane proteins), and these signals can start at nearly the same moment. Since the tissue P systems with parallel rules on channels well capture this phenomenon, we consider it proper to apply it to our model implementation and verification.

References

[1] O. Andrei, G. Ciobanu and D. Lucanu. *Executable Specifications of the P Systems*. In Proceedings of Workshop Membrane Computing, WMC5, LNCS 3365:127-146, 2005.

[2] M. Cavaliere. *Evolution-communication P systems*. In Proceedings of Membrane Computing International Workshop, WMC-CdeA 2002, Curtea de Argeş (Gh. Păun, G. Rozenberg, A. Salomaa, C. Zandron, eds.), LNCS 2597:134-145, 2003.

[3] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. MIT Press, 2001.

[4] Freund, R., Gh. Păun and M.J. Pérez-Jiménez. *Tissue-like P systems with Channel States*. In Proceedings of the Second Brainstorming Week on Membrane Computing (Gh. Păun, A. Riscos, A. Romero, F. Sancho, eds.), Report RGNC 01/04, University of Seville(2004), 206-223, and Theoretical Computer Science, 330:101-116, 2005.

[5] D.T. Gillespie. *Exact Stochastic Simulation of Coupled Chemical Reactions*. Journal of Physical Chemistry, 81:2340-2361, 1977.

[6] R. Krŭger and R. Heinrich. *Model Reduction and Analysis of Robustness for the Wnt/β-catenin Signal Transduction Pathway*. Genome Informatics, 15(1):138-148, 2004.

[7] B. Lustig and J. Behrens. *The Wnt Signaling Pathway and its Role in Tumor Development*. Journal of cancer research and clinical oncology, 129(4):199-221, 2003.

[8] C. Y. Logan and R. Nusse. *The Wnt Signaling Pathway in Development and Disease*. Annual reviews of cell biology, 20:781-810, 2004.

[9] E. Lee, A. Salic, R. Krü ger, R. Heinrich and M. W. Kirschner. *The Roles of APC and Axin Derived from Experimental and Theoretical Analysis of the Wnt Pathway*. PLoS biology, 1(1):116-132, 2003.

[10] V. Manca, L. Bianco and F. Fontana. *Evolution and Oscillation in P Systems: Applications to Biological Phenomena*. Proceedings of the 5th Workshop of Membrane Computing, LNCS 3365:63-84, 2004.

[11] The Maude homepage: http://maude.cs.uiuc.edu/.

[12] M. Clavel et al. *Maude Manual (version 2.2)*. December, 2005.

[13] Martín-vide, C., Gh. Păun, J. Pazos and A. Rodríguez-patón. *Tissue P Systems*. Turku Centre for Computer Science, TUCS Technical Report No.421, Sept. 2001. Also in Theoretical Computer Science, 296(2):295-326, 2003.

[14] R. Nusse. *Relays at the Membrane*. Nature, 438:747-749, 2005.

[15] R. Nusse. *Wnt Signaling in Disease and in Development*. Cell research, 15(1):28-32, 2005.

[16] Gh. Păun. *Computing with Membranes*. Turku Centre for Computer Science-TUCS Research Report No 208, 1998. Also in Journal of Computer and System Sciences, 61(1):108-143, 2000.

[17] Gh. Păun. *Membrane Computing. An Introduction*. Springer Verlag Berlin, 2002.

[18] M. C. Pinto, L. Foss, J. C. M. Mombach and L. Ribeiro. *Modeling and Property Verification of Lactose Operon Regulation*. Proceedings of BSB 2005, LNBI 3594:95-106, 2005.

[19] M.J. Pérez-Jiménez and F.J. Romero-Campero. *A Study of the Robustness of the EGFR Signalling Cascade Using Continuous Membrane Systems*. Proceedings of the First International Work-conference on the Interplay between Natural and Artificial Computation, IWINAC2005, LNCS 3561:268-278, 2005.

[20] A. Regev. *Representation and Simulation of Molecular Pathways in the Stochastic π-Calculus*. Presented at the 2nd Workshop on Computation of Biochemical Pathways and Genetic Networks, 2001.
[21] A. Regev, W. Silverman and E. Shapiro. *Representing Biomolecular Processes with Computer Process Algebra: π-Calculus Programs of Signal Transduction pathways*. American Association for Artificial Intelligence (`http://www.aaai.org`), 2000.
[22] A. Regev, W. Silverman and E. Shapiro. *Representation and Simulation of Biochemical Processes Using the Pi-Calculus Process Algebra*. Proceedings of the Pacific Symposium of Bio-computing 2001(PSB2001), 6:459-470, 2001.
[23] *P Systems Web Page*. http://psystems.disco.unimib.it.
[24] A. Wodarz and R. Nusse. *Mechanisms of Wnt Signaling in Development*. Annual review of cell and developmental biology, 14:59-88, 1988.
[25] Xian Xu. *Tissue P Systems with Parallel Rules on Channels*. Pre-proceedings of the International Conference Bio-Inspired Computing–Theory and Applications (BIC-TA 2006), Wuhan, China.
[26] *The appendix of this paper*. On the website http://basics.sjtu.edu.cn/∼xuxian/.